Predicting Consumption Patterns with Repeated and Novel Events

--Final Project of EECSE6690_001_2018_1 - TOPICS DATA-DRIVEN ANAL & COMP

Chi Zhang, cz2465, cz2465@columbia.edu

Jingyuan Liu, jl4926, jl4926@columbia.edu

Abstract: In the final project, we mainly adopt mixture model to construct the consumption probability matrix. The consumption patterns can be affected by the individual preference and the global popularity. We calculate the multinomial components to represent the probability of these two events based on the training dataset and the mixing weight using EM equations based on the validation dataset. After the matrix has been constructed, we evaluate and compare the performance of three algorithms: NMF, No Smooth NMF, Mixture Model based on the seven different datasets using log-loss and recall@k. It turns out that the mixture model has the best performance and alleviates the over-smoothing problem introduced by NMF. In addition, we also implement the association analysis focused on the data itself.

Key Words: Mixture Model, NMF, Consumption Patterns, Association Analysis

1. Introduction

1.1 Application Area

When users interact with a large amount of items, including the interaction with popular social media website, consuming e-production in the Internet, and geographic location information, etc, the prediction on the consumption patterns becomes more and more important. This kind of recommendation problems or automatic scoring systems are very popular in unsupervised-learning, as they have the essential potential in increasing the commercial value of various relevant industry production.

In general, the consumption patterns of an individual person could be separated into two types:

One is driven by novelty, the other is a mixture of both new items and repeated ones. As for the first, individual person are eager to select items that they have not consumed in the past. Examples of such pattern include purchasing or reading books, watching movies and travelling,

etc. “Datasets exhibiting this pattern have been the primary focus for much of the predictive user modelling work in machine learning and data mining research in recent years.” [1]. The behavior of public selection has significant influences of individual’s prediction in this kind of consumption type.

The second type is a characterized by data mixed both novelty and repeatability, and the repeated ones are always emphasized. In this pattern, “examples include listening to music artists or songs, visiting physical locations, using apps on mobile phones, or purchasing groceries.” [1]. In these kinds of situation, consumers tend to consume repeated items more frequent than novel items. All seven datasets discussed in this paper all maintain this kind of consumption pattern.

1.2 Algorithm

In this paper, we reproduce and explore several popular unsupervised-learning algorithms to exploit and explore the consumption patterns on seven repeated and novel mixed datasets. The algorithms discussed include **Matrix Factorization**, **Mixture Model** and **Association Analysis**. The research process can be separated into three phrases in this paper: First, describing the original data sets and paper that we will reproduce. Then, we will discuss detailed methods and algorithms used in the reproducing, and a comparison study will be processed based on the reproduced experiment and the original method. Finally, two new approaches are proposed and discussed after reproducing results successfully.

2. Original Data Set(s) and Paper(s)

2.1 Description of the paper

The paper we worked on, named “*Predicting Consumption Patterns with Repeated and Novel Events*” [1]. It is a paper that will be published on “IEEE TRANSACTIONS ON KNOWLEDGE AND DATA ENGINEERING, VOL. 30, NO. 8, AUGUST 2018” [1]. In this paper, they try to compare the performance when implementing different algorithms on the same mixed repeated and novel sparse data. They mainly worked on two algorithms and built two models. Non-negative matrix factorization(NMF) and Mixture model are the two algorithms considered here.

And their main result is that Mixture model could get more reliable results than NMF without getting an over-smoothed prediction.

2.2 Description of the data sets

We got the original data set from the "UC Irvine Machine Learning Repository" [2], which contains seven subsets totally. The data are from Twitter, Gowalla, Reddit and Lastfm.

These datasets are all typical consumer-item consumption dataset, which describes the consumption information between users and extremely large amounts of items. The original data are stored in CSV files, containing three columns. Specifically, the first column refers to the index of users, which starts from zero and end in some integer numbers. The second columns refer to the index of items, which also starts from zero and end in some integer numbers. The third column refers to the number of consumption of one user correspond to a certain item. No further information about users and meanings of different items are described in the data, all users and items are defined by integer indexes. In this case, all columns will be described as the same kind of items with different names. However, in different data sets, users and items will be given different meaningful descriptions. Here, we describe each dataset specifically:

tw_oc & tw_ny

In datasets of tw_oc, tw_ny, these two data sets come from Twitter, "Twitter is an online news and social networking service on which users post and interact with messages known as tweets." [3]. "All data was collected between May 2015 and February 2016." [1]. In these data sets, users refer to different tweets, and items refer to geolocation from the Orange County California area and the New York City area separately. And tw_oc contains 13,000 users with 11,000 items, tw_ny contains 30,000 and 11,000 items. The number of consumption corresponds to how many times a certain user has tweeted at a specific geographic location.

go_sf & go_ny

In datasets of go_sf and go_ny, these two data sets come from Gowalla, "Gowalla is a location-based social networking website where users share their locations by checking-in" [4]. All data

are collected from September 2009 to October 2010. In these data, users are the check-ins from the mobile application Gowalla. and items also refer to geographic location information. Data recorded in San Francisco area and New York City area separately. And go_sf contains 2,000 users and 7,000 items, while go_ny contains 1,000 users and 7,000 items. The number of consumption corresponds to how many times a certain user checked_in at a specific geographic location.

reddit_top & reddit_sample

In datasets of reddit_top and reddit_sample, data come from Reddit website. "Reddit is an American social news aggregation, web content rating, and discussion website. Registered members submit content to the site such as links, text posts, and images, which are then voted up or down by other members." [5]. All data contain the posts on Reddit from 2015 to 2016, only subreddits with more than 1,000 subscribers are included in these datasets. In this case, users refer to the registered members of Reddit website, and items are articles, comments and discussions posted on Reddit. The dataset reddit_top has 113,000 users and 21,000 items, while users with less than 10 posts have been excluded here. Another data set reddit_sample is smaller than the former dataset, it refers to a random selection among all users, which contains 20,000 users and 21,000 items. The number of consumption corresponds to "the number of times a registered user posted a comment in subreddit" [1].

lastfm

In the data set of lastfm, data come from a music website "lastfm.com" [6], All data are collected from the year 2007 to 2009. In this case, users refer to the users of lastfm, and items are artists, artists have less than 100 songs are filtered here. The number of consumption here corresponds to "the number of times one user listened to a certain artist" [1]. The dataset lastfm contains 992 users and 15,000 items.

All subsets are separated into three parts, training data, test data and validation data. All data included in the training dataset are defined as the historical information of users, and they are

used to build models by implementing different algorithms. Data in the test data set are defined as the future data of users, they are used to evaluate the performance of the model. Data contained in the validation data set could describe the global performance of the whole set, they are used as compensation for training data when building models.

2.3 Repeated and novel

As discussed in the introduction part, all data discussed in this paper are a mixture of both novel and repeated items. In the specific data sets, the characteristic of repeat means that one user has consumed one item more than once, which the number of consumers is larger than 1. The definition of novel items will consider both the training data and test date, while training data is defined as historical information, and test data is defined as future information. In this case, a novel item refer one item consumed by one user in the future without being consumed by the same user in the previous, which means the number of consumption of this by this user is zero in the training dataset, and the same variable keeps a number larger than zero in the test data set.

3. Reproduction

3.1 Data preprocessing

Because the original data set is too heavy to be processed by our personal laptops and get valid results in limited time. We make a preprocess before we really worked on specific algorithms and take explore our own solutions on these data.

Specifically, we cut off part of the original data set by modifying two thresholds on the index of users and items. And then we evaluate the characteristic of the preprocessed data to make sure whether the preprocess has changed the sparsity, novelty and repeat or not. If the answer if not, we can then rebuild the model and hope to get reasonable reproduced results.

We cut off the seven datasets by setting a threshold number on the index of users and items separately. Specifically, for dataset lastfm, go_ny and go_sf, we only retain the first 500 users and the first 2,000 items, for dataset reddit_sample, reddit_top, tw_oc and tw_ny, we keep the first 1,000 users and the first 1,000 items. After the shrinkage of the data set, we reshape the original n

rows times three columns into a larger sparse matrix. In the matrix, rows refer to users and columns refer to items, each element refers to a specific number of consumption for the correspondent row of users and column of items. For those pairs of users and items have no record in the original dataset, the corresponding elements will be set with an extremely small value 10-16 rather than set to zero. This configuration is to make sure no 00 will appear in the computation of matrix. All measurement are implemented on training data.

3.1.1 Characteristic of Sparsity

We first test how sparse the preprocessed datasets are. In this measurement, we test the **Entries, Density,** and **average number of consumption** of the processed data. Where, the **Entries** refers to the total number of non-zero entries, and the density refer to the value of **Entries** divides all entries which is the size of the whole matrix. And the average number of consumption is the average number of counts for each user.

	Size: $U \times M$	# Entries	Density	$\bar{n}$
redditS	20k x 21k	416k	0.10%	13.2
redditT	113k x 21k	7M	0.29%	41.2
lastfm	992 x 15k	547k	3.86%	27.5
goNYloc	1k x 7k	43k	0.61%	1.5
goSFloc	2k x 7k	71k	0.51%	1.6
twOCloc	13k x 11k	94k	0.07%	3.9
twNYloc	30k x 11k	242k	0.07%	2.3

Table 1.1

	Name	Size: U*M	# Entries	Desity	average count
1	reddit_sample	1000*1000	2146	0.21%	17.9
2	reddit_top	1000*1000	6702	0.67%	50.5
3	lastfm	500*2000	46862	4.69%	22.1
4	go_ny	500*2000	9878	0.99%	1.5
5	go_sf	500*2000	16501	1.65%	1.7
6	tw_oc	1000*1000	6137	0.61%	15.3
7	tw_ny	1000*1000	8257	0.83%	8.3

Table 1.2

Table 1.1 is extracted from the original paper, which describes the characteristic of sparsity of the original data, we rebuild this table using the preprocessed data as Table 1.2. Comparing the result of these two tables. From the first two columns, we have decreased the size of the original data obviously. And because the last two columns of the two tables have almost the same values, the preprocessed data maintains as a sparse data set.

3.1.2 Characteristic of Novelty

In this measurement, the **unique items per user(average)** refers to the average number of consumed items for each user, and the **user-item pairs that are repeats** refers to the average number of consumed repeated items for each users.

	Unique items per user (average)	User-item pairs that are repeats
redditS	20.8	61.1%
redditT	61.9	71.4%
lastfm	547.0	69.5%
goNYloc	43.0	16.3%
goSFloc	35.5	18.3%
twOCloc	7.2	40.4%
twNYloc	8.1	40.7%

Table 2.1

	Name	Unique items per user (average)	User-item pairs that are repeats
1	reddit_sample	3.1	64.7%
2	reddit_top	7.3	74.2%
3	lastfm	95.1	66.7%
4	go_ny	19.9	16.9%
5	go_sf	33.0	18.8%
6	tw_oc	6.1	55.9%
7	tw_ny	8.3	43.9%

Table 2.2

Here, Table 2.1 is extracted from original paper, and Table 2.2 is our reproduced result. The first column is reduced in our result because of the preprocessing, but the second column still keeps the same value like Table 2.1, which means, the preprocessed data set still contain repeated data like the original one does.

3.1.3 Characteristic of Repeat

We want to measure the novelty of preprocessed here. In this situation, **user-item pairs that are new in test data** refers to the number of novel items between training data and test data. While the **user-item events that are new in test data** refer to the number of consumption on novel items between training data and test data.

	User-item pairs that are new in test data	User-item events that are new in test data
redditS	29.8%	11.7%
redditT	20.0%	5.7%
lastfm	21.4%	15.3%
goNYloc	73.2%	62.7%
goSFloc	67.3%	55.0%
twOCloc	45.6%	22.4%
twNYloc	63.5%	42.1%

Table 3.1

	Name	User-item pairs that are new in test data	User-item events that are new in test data
1	reddit_sample	24.2%	8.0%
2	reddit_top	16.3%	4.1%
3	lastfm	33.4%	32.8%
4	go_ny	67.6%	55.5%
5	go_sf	58.7%	48.1%
6	tw_oc	17.0%	5.8%
7	tw_ny	32.2%	9.3%

Table 3.2

Table 3.1 is the result of the paper, and Table 3.2 is our reproduction. In this situation, most of values are keep the same as the original one except that tw_oc and tw_ny are reproduced with a lower value, which might be influenced by the unique data distribution on these two datasets. In this examination, the two columns both consider novel items between training data and test data, however, values of the second columns are all smaller than values of the second column. This indicates a very important characteristic of this kind of consumer-item consumption data:

consumers will consume both repeated and novel items, but they will not consume novel ones frequently. And our reproduction get the same result.

3.1.4 Conclusion

After evaluating the preprocessed data set from the three dimensions, we find that the processed data are still keeping the characteristics of sparsity, repeat and novelty. Then, we could use these new data to build models and implement algorithms.

3.2 Non-negative Matrix Factorization(NMF)

"Matrix factorization (MF) is perhaps the most widely-used approach over the past decade for modeling of sparse user-item consumption data sets" [1]. And NMF is a MF while all elements have non-negative values.

In NMF, the goal is to divide the original matrix $\boldsymbol{M}$ (#users × #items) into two sub-matrix $\boldsymbol{W}$ (#users × rank k) and $\boldsymbol{H}$ (rank k × #items). And we hold the equation:

$$M = WH$$

The rank k defines the sub-variables will be considered in the algorithm, and we hope a low rank k.

In the implementation of NMF algorithm, we run the function NMF which is included in the library NMF [7]**.** This function we used contains 4 main parameters that can be configured:

```
nmf(x, rank, method, nrun)
```

- x: The data will be processed, input as matrix
- rank: The rank k of the output matrix *W* and *H*
- method: The processing algorithm when run NMF
- nrun: The number of run on the input data

In the paper, they set the value of rank as 100 and 500. Because a larger rank will cost more time on computing the result, and we have compress the data in the preprocessing step. We set the rank as 10 in the NMF reproduction step.

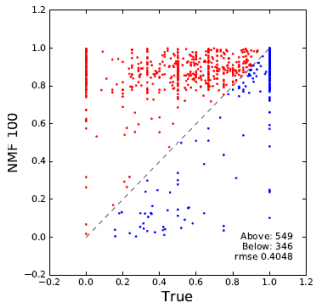


Figure 1.1: go_sf

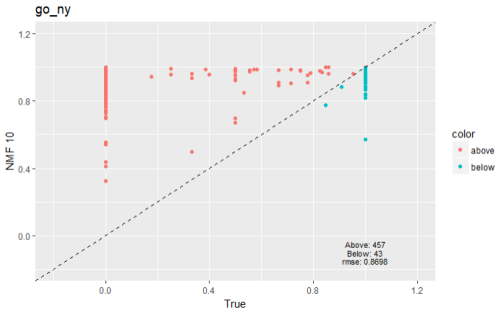


Figure 1.2: go_sf reproduced

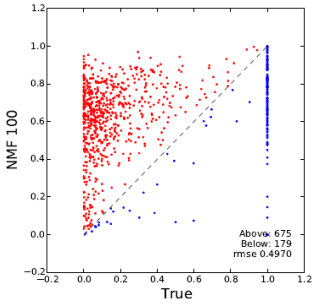


Figure 2.1: lastfm

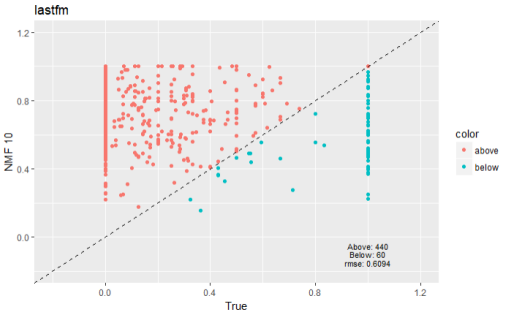


Figure 2.2: lastfm reproduced

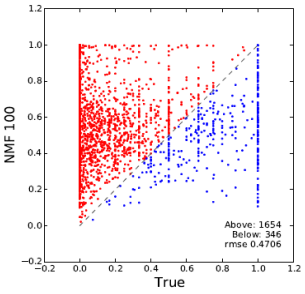


Figure 3.1: reddit_sample

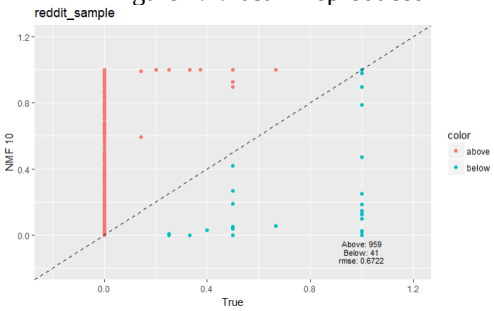


Figure 3.2: reddit_sample reproduced

Figure 1.1, 2.1 and 3.1 present the results of the paper, Figure 1.2, 2.2 and 3.2 are results we reproduced. This kind of scatter plot are one of the important results produced in this paper (Another important result is the evaluation result produced in the Mixture Model part). In each scatters plot, the points refer to users. For each point, the x coordinate value refers to the probability of one user to truly select novel items. This probability is computed by finding the novel items between training data and test data. Then normalize each row of the matrix only contains novel items to get the needed probability. The y coordinate value refers to the probability of one user will select novel items based on the prediction extracted from the model.

In this definition, a good model will generate the prediction that makes more points converge to the middle diagonal line, which means the prediction perform the same consumption as the truth.

However, the results of NMF are not ideal ones obviously, whose points distribute on the whole space. A lot of outliers locate with x coordinate equals to zero, and y coordinate has large positive values. This kind of problem is defined as over-smooth. Because NMF does prediction rely on historical information a lot, it tries to predict each user and item pairs with a large positive value. This characteristic makes MF works well on recommending and evaluating items most of the time. But this paper study on extremely sparse data set, which keeps a lot of elements as zero. To make elements with zero true probability can receive reliable prediction, which should also be zero. The paper explores to using Mixture Model on these datasets, the detailed discussion is covered in the Mixture Model part.

However, the over-smooth problem of NMF could be solved by using a larger rank k, which will encourage individual behavior much more. We also reproduce this result.

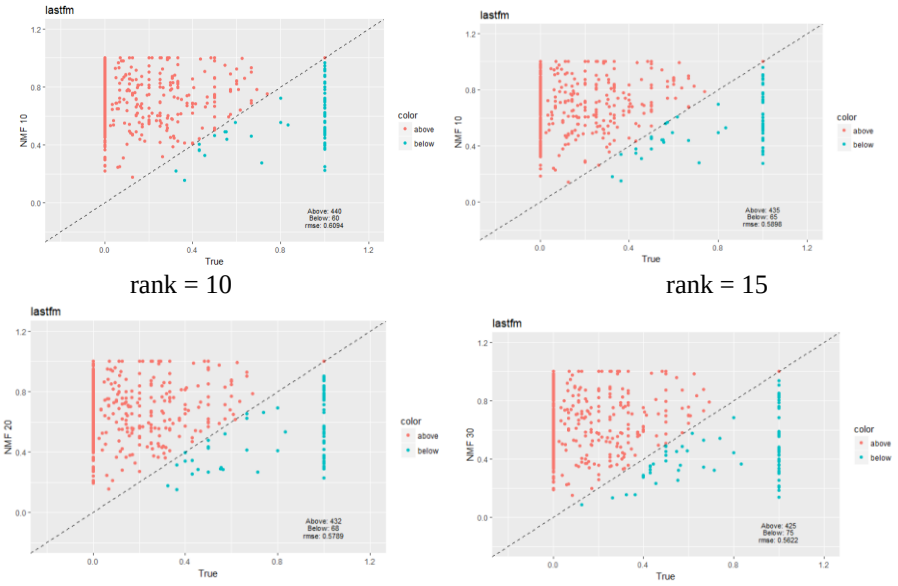


Figure 4: NMF on lastfm with four different ranks

Figure 4 show the result of scatter plots, we try four different rank on the same dataset lastfm. Reading the RMSE values of them, when rank increase from 10 to 30, the value of RMSE decrease from 0.5898 to 0.5622. This means the performance of NMF is increased. However, in the implementation of NMF function, the increase of rank value will lead to an exponential improvement on time cost, besides, we encourage a lower rank in NMF. As a result, increasing

rank is not a appropriate solution on predicting the consumption pattern of repeated, novel and sparse data. This answer is derived both in the paper and our reproduction.

3.3 New Approach 1: Non-smooth NMF

When we read the tutorial of nmf in R, we find the parameter of "method". By modifying method we can choose the algorithm when running nmf. The default set of the method named "brunet", there also exists one selection name "nsNMF", which is the non-smooth algorithm of NMF. Thus, our first exploration is using nsNMF algorithm on NMF method. Here we show the result of the scatter plots of nsNMF.

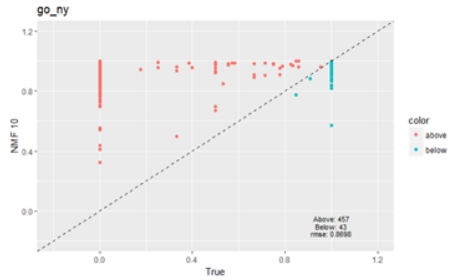


Figure 5.1: go_ny on NMF

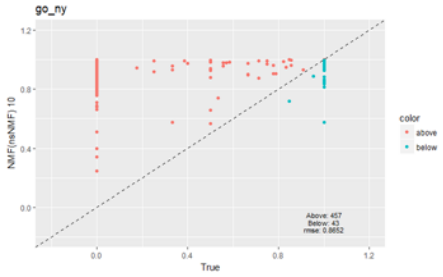


Figure 5.2: go_ny on nsNMF

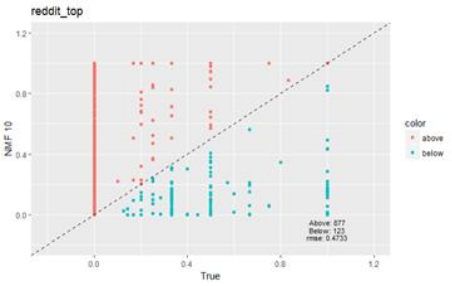


Figure 6.1: reddit_top on NMF

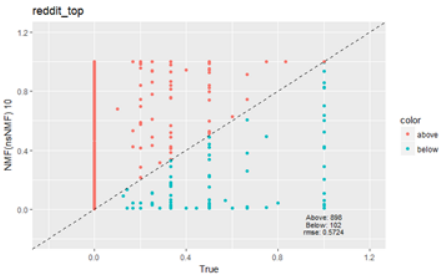


Figure 6.2: reddit_top on nsNMF

Figure 5.1 and 5.2 compare NMF and nsNMF on dataset go_ny, here RMSE decreases from 0.8698 to 0.8652. However, in Figure 6.1 and 6.2, comparison of NMF and nsNMF on dataset reddit_top is shown here, the RMSE increases from 0.4733 to 0.5724. Thus, it is hard to say that the implementation of the nsNMF algorithm will improve the prediction correctness, because the value of RMSE could either increase or decrease, rather than the decrease in all datasets.

But the time cost on nsNMF is much smaller than NMF when they use the same value of rank. Specifically, NMF with default brunet algorithm cost 15 mins to compute a rank 10 matrix, while

NMF with nsNMF algorithm cost 6 mins to compute a rank 10 matrix. This should be the most significant advantage of nsNMF than NMF.

3.4 Mixture Model

To solve the over-smoothing problem, we introduce the No Smooth NMF Algorithm in the section before, however, we have not experienced an obvious improvement. As a result, we adopted the mixture model approaches proposed in the paper, which has been proved to alleviate the over-smoothing problem to a great degree and have better performance in log-loss and recall@k metrics.

Unlike the NMF splitting the consumption patterns matrix into two separate one W and H, the mixture model is to compute the multinomial probability distribution for user u over items in the market. The consumption probability matrix represents the probability of user u purchasing item j. The consumption patterns are commonly affected by two main components: the history consumption records depending on the individual preference and the global popularity of the items. The individual preference is represented by the consumption habits which can be computed given the history records and varies across items and users. The popularity patterns are generated beyond the users' data and vary across items only.

3.4.1 Proof

First, we should prove that the consumption patterns can be predicted through the combination of two multinomial components.

The NMF is a typical unsupervised learning algorithm, however, if we assume there are the probability components underlying, like what has been done in the paper, we can adopt the typical maximum likelihood to optimize the overall parameter $\hat{\theta}_u$ for every user.

$$\theta_{uj}^{MLE} = \frac{n_{uj}}{n_u}, \quad n_u = \sum_{j'} n_{uj'}$$

where n_u represents the total quantity of the items consumed by user u and n_{uj} represents the times of user u consuming item j. Although the parameter estimation is perfect, it is not that

useful to predict the consumption patterns in the future since it leaves a great amount of zero in the matrix.

In order to eliminate the zero in the matrix, we assume there is a prior distribution for θ_u. Same as the assumption made in the paper, we adopt the Dirichlet distribution with parameters $\alpha = [\alpha_1, \alpha_2, \dots, \alpha_M]$. We replace the MLE parameter with the mean posterior estimate (MPE).

$$\theta_{uj}^{MPE} = \frac{n_{uj} + \alpha_j}{n_u + \sum_j \alpha_j}$$

MPE has a better performance in the sparse data matrix. Even though user u has not purchased the item j, we can still have a non-zero estimate for that record thanks to the α_j.

α_j shares among the users, reflecting the global population. As a result, we can assume that each α_j is proportional to $n_j = \sum_u n_{uj}$ since it will not be affected by the individual consumption patterns. We introduce the global parameter η as well to control the smoothing degree, whose value can be determined through the search on validation data. Thus the MPE can be expressed as

$$\theta_{uj}^{MPE} = \frac{n_{uj} + \eta\alpha_j}{n_u + \eta}$$

where

$$\alpha_j = \frac{n_j+1}{n+M}, \qquad n = \sum_j n_j$$

The equation can be rewritten as

$$\theta_{uj}^{MPE} = \gamma_u \frac{n_{uj}}{n_u} + (1 - \gamma_u)\frac{\alpha_j}{\eta}$$

where

$$\gamma_u = \frac{n_u}{n_u + \eta}$$

We can see that the first part is the component reflecting the user's individual preference, derived from the history consumption; the second one is the global popularity only associated with the item j but not with the user u. Thus, the mixture model can be interpreted as the combination of the individual preference and the global popularity with the mixing weights.

The mixing weights here is defined as γ_u, which involves the total count n_u and the prior η. However, the problem here is that the more data we have about user u in the history matrix, the more the history matters. As a result, it will focus more on the repeated consumption behavior and less on the novel consumption patterns.

Therefore, we may have the smoothing problem. Opposite to the one we have with NMF which emphasizes too much on the novel consumption behavior and results to the over-smoothing problem, if we adopt the algorithm above, we may have less exploration in the consumption behavior. Since it is necessary to reflect the explore-exploit personality in persons, we should have another new way to learn the mixing weights while keeping the multinomial components here.

3.4.2 Components Learning

As learned above, we can separate the parameter θ into two parts. Both of them are derived from the train dataset in our project. The first one is the one reflecting the effect of the individual preference on the repeated behavior, derived from the train test as the history data.

$$\theta_{uj}^{I} = \frac{n_{uj}}{n_u}$$

The novel consumption patterns can be assumed as the result of the influence of the global popularity. We write is as

$$\theta_{j}^{P} = \frac{n_j + 1}{n + M}$$

It is clear that θ_{uj}^{I} is related to both user u and item j, however, the θ_{j}^{P} is only associated with the item j.

3.4.3 Mixing Weights Learning

To construct the consumption probability matrix, it is necessary to calculate the mixing weight for each user to show the probability of the user consuming the repeated or the novel items.

Global Mixing Weight

The mixing weights learning process has been split into two parts. The first one is to learn the global mixing weight, which represents the preference globally. π represents that the probability of the users preferring the repeated consumption patterns globally, while $1 - \pi$ represents the probability of the users more willing to consume novel items. Moreover, it is used to model the preference for each user π_u with a binomial distribution as the Beta Prior.

$$\mathfrak{B}(\beta^I, \beta^P) = \mathfrak{B}(\pi \times \bar{n}, (1 - \pi) \times \bar{n})$$

For the users with no records in the validation, when we compute the prediction matrix with test data, we can assign the π_u as π. As a result, we are able to predict the consumption probability even the users do not have any historical data records.

We adopt the EM equations to calculate the π. Let D_V be the amount of the times in the validation dataset user u purchased item j, we can write the equation as

$$p(D_V|\theta, \pi) = \prod_{j=1}^{M} p(j|\theta, \pi)^{n_{uj}}$$

$$= \prod_{j=1}^{M} (\pi_u \, p^I(j|\theta^I) + (1 - \pi_u) p^P(j|\theta^P))^{n_{uj}}$$

For the E-step, we set the probability that the consumption is totally generated by the individual preference.

$$z_j = \frac{\pi p^I(j|\theta^I)}{\pi p^I(j|\theta^I) + (1 - \pi) p^P(j|\theta^P)}$$

And the correspondent Beta prior as

$$\beta^I = \pi \times \bar{n}$$

$$\beta^P = (1 - \pi) \times \bar{n}$$

For the M-step, we update the value of π by summing all points and normalizing the value to sum to 1.

$$\pi^{(t+1)} = \frac{\sum_{u=1}^{N} \sum_{j=1}^{M} n_{uj} z_j + \beta^I - 1}{\sum_{u=1}^{N} \sum_{j=1}^{M} n_{uj} + \beta^I + \beta^P - 2}$$

Such EM algorithm converges only after a few iterations. In the coding, we set the iteration times

as 10, enough for the result to converge.

3.4.4 Individual Weight learning

The calculation for the individual mixing weight adopts the same EM equation. The difference is that here we use the data for the user u only but not the overall data. Thus, we can rewrite the calculation equations as

$$z_{uj} = \frac{\pi_u p^I(j|\theta^I)}{\pi_u p^I(j|\theta^I) + (1 - \pi_u) p^P(j|\theta^P)}$$

$$\pi_u^{(t+1)} = \frac{\sum_{j=1}^{M} n_{uj} z_{uj} + \beta^I}{\sum_{j=1}^{M} n_{uj} + \beta^I + \beta^P}$$

In the coding, we find that the β^I and β^P here are both less than 1, as a result, we eliminate the -1 and -2 parts in the equation to make sure the π_u is non-negative. We also set the iteration times here as 10.

Consumption Probability Matrix

After we get the mixing weights and two multinomial components, we can construct the consumption probability matrix for every user u as

$$P_{uj} = \pi_u \theta_{uj}^I + (1 - \pi_u)\, \theta_j^P$$

where π_u is the mixing weight we learnt about user u, θ_{uj}^I is the component indicating individual preference of user u for item j and θ_j^P represents the global popularity of item j.

3.4.5 Over-Smoothing Solution

We construct the scatter plot for each dataset similarly to what we do for NMF and No Smooth NMF using consumption probability matrix we compute above. The x-axis represents the probability of novel consumption patterns in the test dataset compared to the train dataset. The y-axis represents the probability of novel patterns in the consumption probability matrix compared to the train dataset.

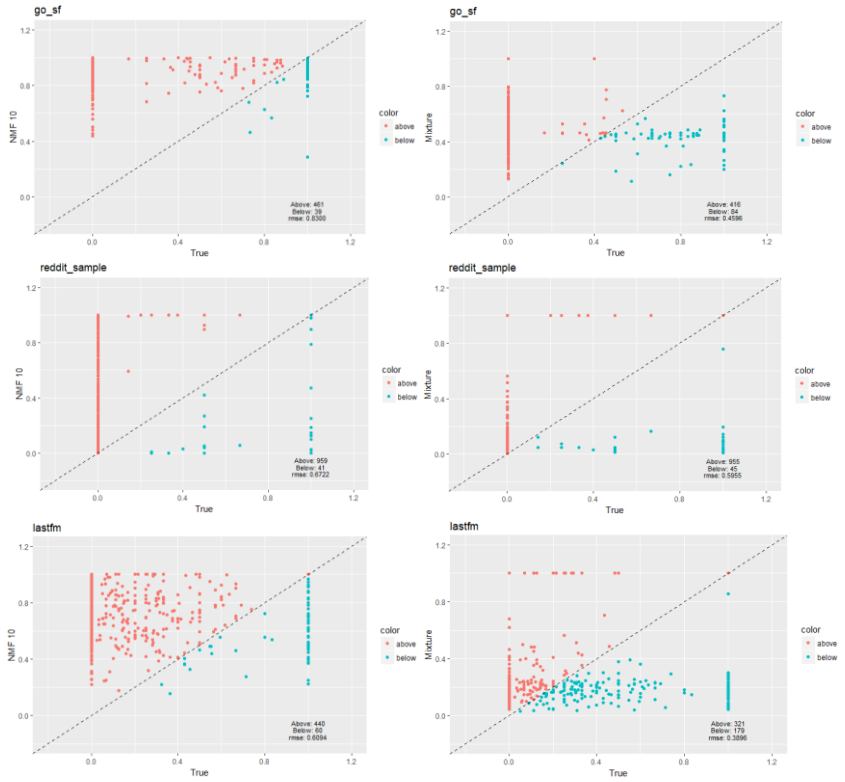


Figure 7: Scatter plots comparison between NMF and Mixture Model of go_sf, reddit_sample and lastfm

The left figures are the scatter plots computed from NMF and the right figures are computed from the mixture model. We can clearly see that when x is close to 0, the y values in the mixture model are obviously smaller than the ones in the NMF figures. Moreover, the y values there converge to the diagonal line means the over-smoothing problem has been alleviated a lot with the mixture model.

3.4.6 Evaluation

Since we have already constructed the consumption matrix with the mixture model, what we do in this section is the evaluation. We adopt two metrics to evaluate the performance, one is the log loss function, the other is the recall@k.

Log Loss

Log loss is used commonly to evaluate the machine learning algorithms, especially for the probability matrix. It takes the negative of the average log probability for each consumption pattern.

$$-logP = -\frac{1}{N_{te}}\sum_u \sum_j n_{uj} logP(j|u)$$

We adopt the test dataset for the evaluation since the component and the mixing weight are learnt from the train dataset and the validation dataset. If the user u consumes the item j in the test dataset and we predict such consumption patterns successfully, we can have lower negative log loss. For the data not in the test dataset, no matter what its probability is in the consumption probability matrix we construct, it will be recorded as 0. As a result, the smaller the value is, the better the algorithm performs.

We compute the log-loss of seven datasets using NMF, No Smooth NMF and the mixture model.

	NMF 10	NMF ns 10	Mixture
redditS	2.276	inf	0.881
redditT	1.694	inf	0.815
lastfm	6.814	inf	5.262
goSFloc	8.85	inf	5.766
goNYloc	12.1	inf	5.702
twOCloc	3.265	inf	1.442
twNYloc	2.386	inf	1.278

It is clear that among the three algorithms, the mixture model has the smallest log-loss value, which means it has the best performance and the improvement is obvious.

Recall@k Metrics

The recall@k metric is used to evaluate the ability of the algorithm to assign the high probability to the items, where k means the top k items we will select from the consumption probability matrix.

$$\text{Re}call@k = \frac{1}{N_{te}}\sum_u \sum_j \frac{n_{uj} 1\{rank(u,j) \leq k\}}{\sum_{j'} n_{uj'}}$$

We first rank the probability of user u consuming item j in the descending order and choose the top k items and take the amount of consumption in the test data into account. The higher the scores of the recall@k metric, the better the algorithm is. However, compared to the log-loss function, it might not be a better evaluation method since it is only focused on the top k items

rather than focus on all items.

We compute the recall@k of seven datasets using NMF, No Smooth NMF and the mixture model.

	NMF 10	NMF ns 10	Mixture
redditS	0.0713	0.0647	0.073
redditT	0.0183	0.0139	0.0187
lastfm	0.0054	0.0049	0.007
goSFloc	0.0675	0.0624	0.074
goNYloc	0.114	0.108	0.143
twOCloc	0.2	0.199	0.208
twNYloc	0.16	0.156	0.17

It is clear that among the three algorithms, the mixture model has the largest recall@k value where we set k as 100 in our work, which means it has the best performance.

3.5 Association Analysis

It is hard for us to find a novel algorithm to improve the overall performance, we have tried the No Smooth NMF, but there is little improvement as mentioned above. As a result, we try to focus more on the data itself. Thus, we perform the association analysis to figure out the relationship between the items that may be in the same basket.

Here are some concepts in the association analysis that we mention in our work.

We can write the association rule as the form as

$$A \rightarrow B$$

where A and B are disjoint.

The strength can be measured as support, confidence and lift. Support determines the frequency of the given dataset, confidence is the frequency of the item B given the presence of item A and lift is the increase in the confidence of the item B given item A compared to the probability of item B in the basket.

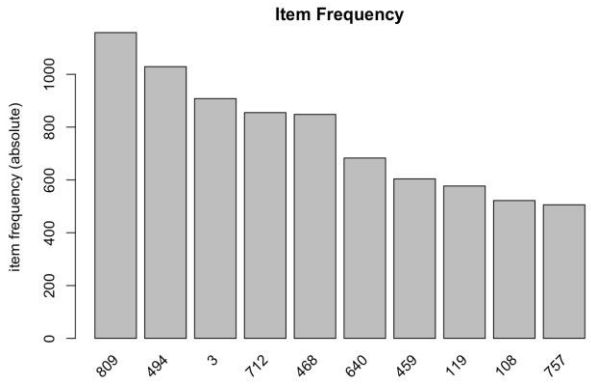
$$Support, s(A \rightarrow B) = \frac{p(X \cup Y)}{N}$$

$$Confidence, c(A \rightarrow B) = \frac{p(X \cup Y)}{p(X)}$$

$$Lift, l(A \rightarrow B) = \frac{c(A \rightarrow B)}{p(B)} = \frac{p(X \cup Y)}{p(X)}$$

We take the location data in San Francisco, names as go_sf here for the example.

We first compute the general item frequency to see the items consumed most often by the users.



In the application of the rules, we set the threshold as 0.005, which means the frequency of the item less than 0.005 will be ignored for the next-step search. The results shown in the Figure 8 are ranked through the confidence and the top 6 items have been shown here. The items in the Figure 9 are ranked through the lift.

```
    lhs                rhs    support     confidence lift      count
[1] {494,685,1029} => {640} 0.005198358 0.9500000  10.167643 38
[2] {119,494,685}  => {640} 0.005471956 0.9302326   9.956076 40
[3] {129}          => {712} 0.008344733 0.9104478   7.784062 61
[4] {129,640}      => {712} 0.005471956 0.9090909   7.772461 40
[5] {494,685,712}  => {640} 0.006019152 0.8979592   9.610661 44
[6] {494,685}      => {640} 0.011901505 0.8877551   9.501449 87
```

Figure 8

```
    lhs                rhs    support     confidence lift       count
[1] {385}          => {367} 0.005198358 0.8444444  121.037037 38
[2] {640,1517}     => {685} 0.006566347 0.8000000   16.154696 48
[3] {494,685,1029} => {640} 0.005198358 0.9500000   10.167643 38
[4] {119,494,685}  => {640} 0.005471956 0.9302326    9.956076 40
[5] {494,685,712}  => {640} 0.006019152 0.8979592    9.610661 44
[6] {494,685}      => {640} 0.011901505 0.8877551    9.501449 87
```

Figure 9

Take the third item whose index is 129 in the Figure8 as the example, given the item 129 is in the basket, the probability of item 712 in the same basket is 0.91, which has been lifted 7.78 compared to the original result.

4. Conclusion and Discussion

Although NMF is a popular algorithm which can be applied to predict the consumption patterns, it can have the avoidable over-smoothing problem. We first try the No Smooth NMF algorithm, however, we can not get a better improvement. But for the No Smooth NMF algorithm, we can clearly see the decrease in the execution time, which means we can spend less time running the algorithm while getting the same results.

Based on the results, we mainly adopt the mixture model to predict the consumption patterns matrix based on the repeated and novel events. There are main components in the algorithm, one is the individual preference, the other is the global popularity. Based on the train dataset, these multinomial components can be computed. Another step in this algorithm is to calculate the mixing weight using EM algorithm. We adopt the same equation for the calculation, and the difference is that for the global mixing weight, we use all data; while for the individual mixing weight, we use the data only for user u. The data for the mixing weight computation is from the validation dataset.

In addition, we also evaluate three algorithms using log-loss and the recall@k metrics. The clear improvement of the mixture model can be observed. However, there is still the difference between the value we get and the lower bound, which means the algorithm still needs improvement.

Furthermore, since t is hard for us to find a novel algorithm to improve the overall performance, we try to focus more on the data itself. Thus, we perform the association analysis to figure out the relationship between the items that may be in the same basket. The results have been shown above.

In the future, we hope to reproduce the results using the whole original data set rather than using a preprocessed version with more powerful devices. And we hope to focus on the performance of the NMF, we will have a deep view on the inner algorithms used in NMF function and extract an more efficient method to alleviated the problem of over-smooth.

Our source code has been uploaded to GitHub [8].

Reference

[1] Dimitrios Kotzias, Moshe Lichman, Padhraic Smyth. Predicting Consumption Patterns with Repeated and Novel Events, IEEE TRANSACTIONS ON KNOWLEDGE AND DATA ENGINEERING, VOL. 30, NO. 8, AUGUST 2018, https://dkotzias.com/papers/repeat.pdf

[2] Repeat Consumption Matrices Data Set, UCI Machine Learning Repository, https://archive.ics.uci.edu/ml/datasets/Repeat+Consumption+Matrices

[3] Twitter wiki, https://en.wikipedia.org/wiki/Twitter

[4] Gowalla official website, https://snap.stanford.edu/data/loc-gowalla.html

[5] Reddit wiki, https://en.wikipedia.org/wiki/Reddit

[6] Lastfm official website, https://www.last.fm/zh/

[7] NMF tutorial, R_studio, https://cran.r-project.org/web/packages/NMF/vignettes/NMF-vignette.pdf

[8] GitHub link of this final project, https://github.com/Debug1995/Consumption-Analysis